Artificial Neural Networks. A New Approach to Modelling Interregional Telecommunication Flows

Manfred M. Fischer and Sucharita Gopal

Vienna University of Economics and Business, Boston University

1994

WSG Diskussionspapiere

WSG 38/94

Artificial Neural Networks. A New Approach to Modelling Interregional Telecommunication Flows

Manfred M. Fischer and Sucharita Gopal

Institut für Wirtschafts- und Sozialgeographie

Wirtschaftsuniversität Wien

Department of Economic and Social Geography

Vienna University of Economics and Business Administration

WSG Discussion Papers

Abteilung für Theoretische und Angewandte Wirtschafts- und Sozialgeographie
Institut für Wirtschafts- und Sozialgeographie
Wirtschaftsuniversität Wien

Vorstand: o.Univ.Prof. Dr. Manfred M. Fischer
A - 1090 Wien, Augasse 2-6, Tel. (0222) 313 36 - 4836

Redaktion: Mag. Petra Staufer

WSG 38/94

Artificial Neural Networks. A New Approach to Modelling Interregional Telecommunication Flows

Manfred M. Fischer and Sucharita Gopal

WSG-Discussion Paper 38

March 1994

Gedruckt mit Unterstützung
des Bundesministerium
für Wissenschaft und Forschung
in Wien

WSG Discussion Papers are interim reports presenting work in progress and papers which have been submitted for publication elsewhere.

ISBN 3 85037 043 7

Abstract

During the last thirty years there has been much research effort in regional science devoted to modelling interactions over geographic space. Theoretical approaches for studying these phenomena have been modified considerably. This paper suggests a new modelling approach, based upon a general nested sigmoid neural network model. Its feasibility is illustrated in the context of modelling interregional telecommunication traffic in Austria and its performance is evaluated in comparison with the classical regression approach of the gravity type. The application of this neural network approach may be viewed as a three-stage process. The first stage refers to the identification of an appropriate network from the family of two-layered feedforward networks with 3 input nodes, one layer of (sigmoidal) intermediate nodes and one (sigmoidal) output node (logistic activation function). There is no general procedure to address this problem. We solved this issue experimentally. The input-output dimensions have been chosen in order to make the comparison with the gravity model as close as possible. The second stage involves the estimation of the network parameters of the selected neural network model. This is performed via the adaptive setting of the network parameters (training, estimation) by means of the application of a least mean squared error goal and the error back propagating technique, a recursive learning procedure using a gradient search to minimize the error goal. Particular emphasis is laid on the sensitivity of the network performance to the choice of the initial network parameters as well as on the problem of overfitting. The final stage of applying the neural network approach refers to the testing of the interregional teletraffic flows predicted. Prediction quality is analysed by means of two performance measures, average relative variance and the coefficient of determination, as well as by the use of residual analysis. The analysis shows that the neural network model approach outperforms the classical regression approach to modelling telecommunication traffic in Austria.

1. Introduction

Telecommunication like transportation interactions, take place over geographic space. But in contrast to the vast literature devoted to the analysis of region-to-region travel demand (see Fischer 1993), relatively little research effort has been directed towards region-to-region telecommunication demand. One reason for this deficiency in research is that telecommunication data are either not available as in the case of most European countries or viewed as proprietary by the private companies (as in the case of US) which provide the telecommunication services (Guldmann 1992). A better understanding of the spatial structure of telecommunication interactions is becoming more and more important especially in the context of Europe where the telecommunication sector is increasingly coming under debate due to deregulation trends. The knowledge of telecommunication linkages might also assist to clarify issues such as the complementarity and substitution effects between transportation and telecommunication (Salomon 1986) and the value of telecommunication technologies as patterns of regional development (see Gillespie and Williams 1988).

The *general purpose* of this paper is to set out an (artificial) neural network (connectionist) approach to modelling interregional telecommunication interactions and, thus, to contribute to the debate on the feasibility of neural network computing (neurocomputing) in geography and regional science (Openshaw 1992, Fischer and Gopal 1993). The *specific objectives* may be stated as follows: first, to develop and estimate a neural net telecommunication flow model with explicit treatment of geographical distance, second, to evaluate its prediction quality in relation to the classical regression model of the gravity type.

The general model suggested is based upon the family of two-layered feedforward neural networks with sigmoidal non-linearities, the most tractable and most prominent family of artificial neural net models (see Hecht-Nielsen, 1990). The value of this approach to modelling interregional telecommunications will be illustrated on the basis of an application using a new telecommunication data set of Austria. The application of the approach may be viewed as a three-stage process. The first stage refers to the identification of an appropriate model candidate from the family of two-layered feedforward networks. The input-output dimensions are chosen so as to make the comparison to the gravity model as close as possible. The second stage involves the estimation of the network parameters of the selected neural network model. This is performed via the adaptive setting of the network parameters by means of the application of a least mean squared error goal and the error back propagating technique, a recursive learning procedure using a gradient search to minimize the error goal. Particular emphasis is laid on the sensitivity of the network performance to the choice of the initial

network parameters as well as the problem of overfitting. The final stage of applying the approach refers to the testing of the interregional teletraffic flows predicted. Prediction quality is analyzed by means of performance measures as well as by the use of residual analysis and compared with gravity model predictions.

The remainder of this paper is organized as follows. Section 2 briefly characterizes the classical regression model of the gravity type that is used as a benchmark model in this study, then proceeds to outline the neural networks approach by considering it as a three stage process, and lastly, describes the estimation process based upon a supervised learning algorithm. The experimental environment and the results of model identification, estimation and testing are discussed in relation to the classical telecommunications flow model in section 3 . Conclusions and areas for further research are outlined in section 4.

2. The Methodological Framework

The literature on telecommunication flow modelling is primarily based on the conventional regression approach of the gravity type (Pacey 1983, Rietveld and Janssen 1990, Rossera 1990, Fischer et al. 1992, Guldmann 1992), and thus lies in the tradition of the broader and well-established framework of spatial interaction modelling in geography and regional science (see for example, Fotheringham and O'Kelly 1989). This conventional model approach serves as a benchmark for evaluating the performance of the alternative, the neural net approach. We first briefly characterize the conventional modelling approach, and then the general neural network model.

2.1 The Gravity Model as Benchmark

Let T_{rs} denote the intensity of telecommunication from region r to region s (r, s =1,...,n), measured in terms of erlang or minutes. Then T_{rs} is called interregional traffic, if $r \neq s$, and intraregional traffic, if r=s.

The conventional approach applied to the problem of modelling telecommunications traffic belongs to the class of (unconstrained) gravity models. It is usually assumed (see Rietveld and Janssen 1990, Fischer et al. 1992) that

$$T_{rs}^{conv} = G\ (A_r,\ B_s,\ F_{rs}) \qquad r, s = 1, \ldots, n \qquad (1)$$

with

A_r a factor associated with the region r of origin and representing the intensity of telephone communication generated by this region,

B_s a factor associated with the region s of destination and representing destination-specific pull factors or the degree to which the in-situ attributes of a particular destination attract telephone communication traffic, and

F_{rs} a factor associated with the origin - destination pairs (r, s) representing the inhibiting effect of geographic separation from region r to region s.

A_r and B_s are called mass or activity variables, F_{rs} is termed separation variable, provided it is specified in a way such that higher values imply less telecommunications traffic. In this approach, the specific form of the function to be fitted to the data is first chosen and then the fitting according to some error criterion (usually mean squared error) is carried out.

G is usually specified in such a way that the interregional telecommunication flow model reads as follows:

$$T_{rs}^{conv} = K \; A_r^{\alpha_1} \; B_s^{\alpha_2} \; F_{rs}\,(D_{rs}) \qquad r, s = 1, \ldots, n \qquad (2)$$

with

$$F_{rs}\,(D_{rs}) = D_{rs}^{\alpha_3} \qquad r, s = 1, \ldots, n \qquad (3)$$

where T_{rs}^{conv} denotes the intensity of telecommunication from r to s predicted by (2) - (3). A_r and B_s represent the potential pool of calls in region r and the potential draw of calls in region s, respectively. We have decided to use gross regional product as a measure for A_r and B_s. Gross regional product as a proxy of economic activity and income is relevant for both business and private phone calls (see Rietveld and Janssen 1990). D_{rs} denotes distance from region r to region s. K is a scale parameter (constant), α_1, α_2 and α_3 are parameters to be estimated. n denotes the number of telecommunication regions.

The usual strategy to estimating (2) - (3) is to assume that a normally distributed multiplicative error term ε_{rs} applies, i.e. $\varepsilon_{rs} \sim N(o,\sigma)$ independently of A_r, B_s and D_{rs}. In this case, OLS-regression can be applied after a logarithmic transformation yielding:

$$\ln T_{rs}^{conv} = \ln K + \alpha_1 \ln A_r + \alpha_2 \ln B_s + \alpha_3 \ln D_{rs} + \tilde{\varepsilon}_{rs} \qquad r, s = 1, \ldots, n \qquad (4)$$

OLS-estimation leads to unbiased and consistent estimates $\hat{\alpha}_1$, $\hat{\alpha}_2$ and $\hat{\alpha}_3$. This model version is usually called log-normal gravity model and will be used as benchmark to

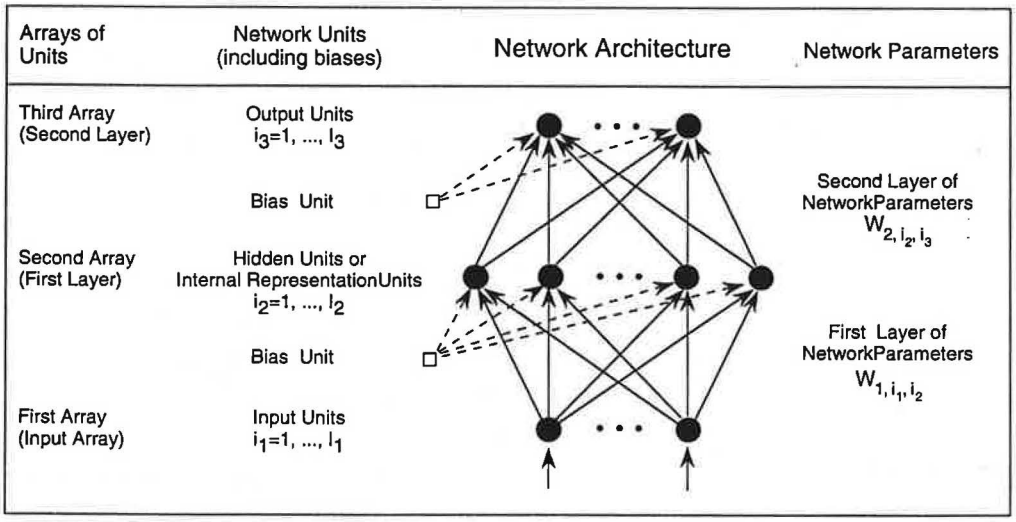
evaluate the relative efficiency of the neural net model to be developed in the sequel. It is worthwhile to note that there are some problems with this model. For example, it is based on the questionable assumption that the variances of the error terms are identical, and it cannot be used when some of the interaction flows are zero.

The standard method for assessing the goodness of fit of this regression model is through the use of R^2, the coefficient of determination. It is worthwhile to note that the OLS-estimator of the log-normal gravity model minimizes the sum of squared residuals and, thus, automatically maximizes R^2. Consequently, maximization of R^2, as a criterion for the OLS-estimator, is formally identical to the least squares criterion. An unsatisfactory fit in terms of R^2 may result from the failure to include all relevant explanatory variables, such as for example, barrier effects impeding the continuous flow pattern in space (see Fischer et al. 1992).

2.2 The General Neural Network Model

The general neural network approach set out in this section is based upon the family of two-layered feedforward neural networks with sigmoidal processing units. The two-layered feedforward neural network is a particular neural network architecture characterized by a hierarchical design consisting of I_1 input nodes, one layer of I_2 intermediate (hidden) nodes and I_3 output nodes as displayed in figure 1.

Figure 1: The General Two-Layer Feedforward Neural Network Model



The input units, $i_1=1, ..., I_1$, send (continuous-valued) signals Y_{1, i_1} to intermediate (hidden) units, indexed by $i_2=1, ..., I_2$, that process them in some characteristic way (linear summation of input signals multiplied by the corresponding network weights) and produce a bounded output signal Y_{2, i_2} that is sent via the second layer of connections to

the output units, indexed by $i_3=1, \ldots, I_3$, which now treat the hidden units as inputs and produce the network output Y_{3,i_3}. Note also the presence of bias units. These output a fixed unit value and can be viewed of as constant terms in the equations defining the process carried out by the network. The network has two layers of unidirectional connections with which weights (parameters) are associated. The first layer of weights connects the input units to the hidden units, and the second layer the hidden units to the output units. The architecture is feedforward because signals flow in only one direction. The fundamental characteristic of this network architecture is that there are no connections leading from an unit to units in previous arrays nor to the other units in the same array, nor to the units more than one array ahead. The arrays of nodes are connected in such a way that the units are connected to every unit in the next array.

The general two-layer neural network model, denoted by $\aleph_{I_1, I_2, I_3}$ or $(I_1{:}I_2{:}I_3)$, may be described as follows:

$$X_{2,i_2} = \sum_{i_1=1}^{I_1} W_{1,i_1,i_2}\, Y_{1,i_1} + W_{1,I_1+1,i_2} \qquad i_2=1, \ldots, I_2 \qquad (5)$$

$$Y_{2,i_2} = f(X_{2,i_2}) \qquad i_2=1, \ldots, I_2 \qquad (6)$$

$$X_{3,i_3} = \sum_{i_2=1}^{I_2} W_{2,i_2,i_3}\, Y_{2,i_2} + W_{2,I_2+1,i_3} \qquad i_3=1, \ldots, I_3 \qquad (7)$$

$$Y_{3,i_3} = f(X_{3,i_3}) \qquad i_3=1, \ldots, I_3 \qquad (8)$$

with

I_j number of processing units of the j–th array (j=1, 2, 3),

i_j indices associated with the j–th array of units (j=1, 2, 3),

Y_{j,i_j} (continuous–valued) output of the units i_j of the j–th array (j=1, 2, 3),

X_{j,i_j} (continuous–valued) input to the processing unit i_j belonging to the j–th array (j=2, 3), note: $X_{1,i_1} \equiv Y_{1,i_1}$,

$W_{k,i_k,i_{k+1}}$ weights (connection weights, parameters) of the k–th array between the k–th and (k+1) –th array of units with k=1: connection weights from the hidden to the output layer, k=2: connection weights from the input array to the hiden layer (note: $W_{k,i_k,i_{k+1}}$ is a positive number if unit i_k excites unit i_{k+1}, and a negative number if unit i_k inhibits unit i_{k+1}),

W_{1,I_1+1,i_2} bias unit of hidden units,

W_{2,I_2+1,i_3} bias unit of the output units.

where the (transfer) function f is the following logistic function of hidden and output units, i_2 and i_3:

$$f(X_{k,i_k}) = \frac{1}{1 + \exp(-a\ X_{k,i_k})} \qquad k=2,3 \qquad (9)$$

(9) scales the activation of a unit sigmoidally between 0 and 1. Strict monotonocity implies that the activation derivative of f is positive: $df/dX_{k,i_k} = a\ f(1-X_{k,i_k}) > 0$. This property increases the networks computational richness and facilitates noise suppression. The slope of the sigmoid, a, can be absorbed into weights and biases without loss of generality and is set to one.

The parameters of (5) - (8) $\{W_{k,\,i_k,\,i_{k+1}};\ i_k=1,\ \ldots,\ I_k;\ k=1,\ 2\}$ and the adaptable biases $\{W_{1,\,I_1+1,\,i_2},\ W_{2,\,I_2+1,\,i_3}\}$, briefly $W \equiv \{W_{k,\,I_k+1,\,i_{k+1}};\ k=1,\ 2\}$ have to be estimated. The weights correspond to a point ω in the $K = (\sum_{k=1}^{2} I_{k+1}\ (I_k+1))$-dimensional euclidean space $\Re^K$. For every point ω in the network configuration space $\Omega \subset \Re^K$, the network model (5)-(9) is a realization of a deterministic mapping from an input, say $x \in X \subset \Re^{I_1}$, to an output, say $y \in Y \subset \Re^{I_3}$ (see Levin, Tishby, and Solla 1990). We denote this mapping by $y = F_\omega(x)$, $F_\omega: \Re^{I_1} \times \Omega \rightarrow \Re^{I_3}$.

Inserting (5) - (7) and (9) into (8) describes the general neural net model in the following compact form, for $i_3 = 1,\ldots,\ I_3$.

$$Y \equiv Y_{3,i_3} = \left[1 + \exp\left[-\left(\sum_{i_2=1}^{I_2} W_{2,i_2,i_3}\left(1 + \exp\left(-\left(\sum_{i=1}^{I_1} W_{1,i_1,i_2}\ Y_{1,i_1} + W_{1,I_1+1,i_2}\right)\right)\right)^{-1} + W_{2,I_2+1,i_3}\right)\right]\right]^{-1}$$

$$\equiv F_\omega(X) \qquad (10)$$

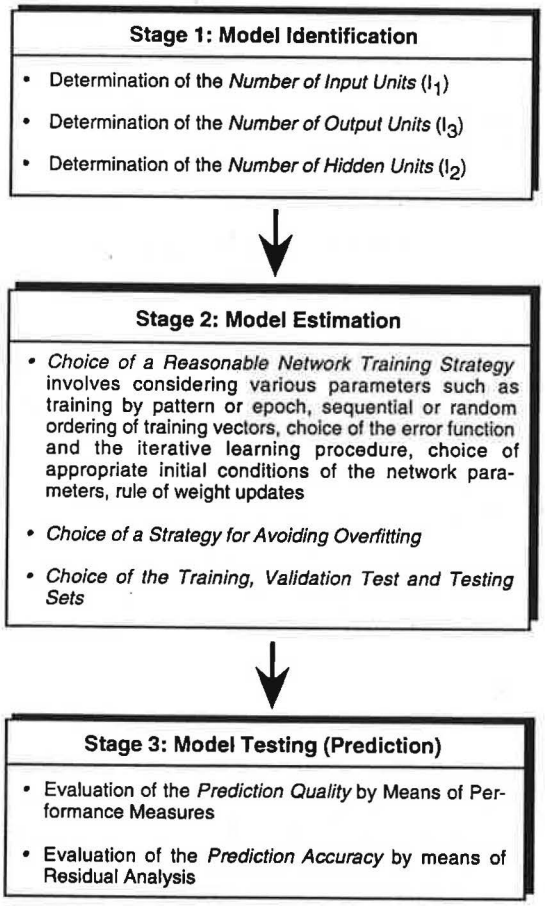
The general model defined by (10) can be viewed as an input-output model $Y=F_\omega(X)$ or as a non-linear regression function with quite a specific form. Its usefulness as a flexible functional form hinges on the nature of the set of mappings from input to output spaces that it can approximate (White 1989). Recently, Hornik, Stinchcombe and White (1989) and Cybenko (1989) have shown that networks of this type, with one layer of hidden units, can approximate any continuous input-output relation of interest to any desired degree of accuracy, provided sufficiently many hidden units are available (see also Hecht-Nielsen 1990). This result shows the atttractivity of (10).

2.3 Neural Network Modelling as a Three-Stage Process

The application of the general model (10) may be considered as a three stage modelling process (see figure 2). The *first stage* refers to the identification of a specific model from $\aleph_{I_1, I_2, I_3}$ for the problem at hand and involves the determination of I_1 , I_2 and I_3 . In order to make the comparison as close as possible with the gravity model, we identify the following subclass $\aleph_{I_1, I_2, I_3}$ defined as follows(see figure 3):

- I_1=3 input units corresponding to the three independent variables A_r, B_s, and D_{rs} in (4) and
- I_3=1 output unit producing the neural network (telecommunication flow) prediction $Y_{3,1}$ (=T_{rs}^{neur}) to be compared with T_{rs}^{conv},
- The number I_2 (hereafter abbreviated as "I") of (sigmoidal) hidden units is a priori unknown.

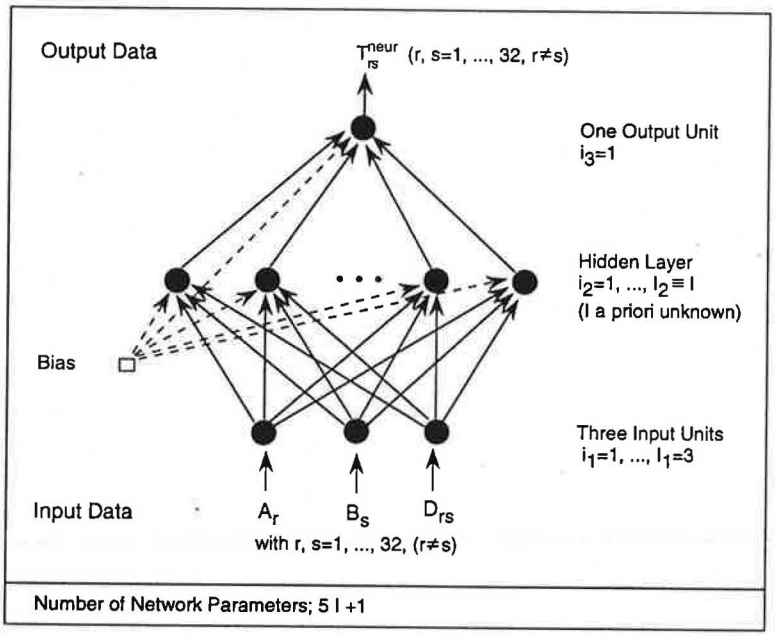
Figure 2: Neural Network Modelling as a Three-Stage Process



There is no general procedure to determine "I" exactly. Consequently, identification and estimation overlap in the model building process. Here we employ the estimation procedure to carry out part of the identification and determine "I" experimentally by means

of performance measures which indicate that specific model worthy of further investigation.

Figure 3: A Subclass of the General Two-Layer Feedforward Neural Network Model to Modelling Telecommunications over Space



The *second stage* refers to the estimation of the network parameters. Once an appropriate network model has been chosen, much of the effort in neural network building concerns the design of a reasonable network training (estimation) strategy. This necessitates engineering judgement in considering various parameters: training by pattern or epoch, sequential or random ordering of training vectors, choice of the error goal, and the iterative learning procedure, choice of appropriate initial conditions on the network parameters. The non-linear character of (10) necessitates a computationally intensive iterative solution of determining the network parameters. As shown in what follows in 2.4, the estimation of the network parameters is performed via adaptive setting by means of the application of a least mean squared error goal and the error back propagating technique, a recursive learning procedure using gradient search to minimize the error goal. A serious problem in model estimation is the problem of overfitting which is especially serious for noisy real world data of limited file length (see 3.3 for more details).

The *final stage* involves the prediction of the interregional telecommunication flows and evaluating the model's testing performance by means of performance measures and by the use of residual analysis, in relation to the benchmark model.

2.4 The Training or Estimation Process

The problem of finding a suitable set W of parameters that approximate an unknown input-output relation $Y = F(X)$ is solved using a supervised learning algorithm. Supervised learning requires a training set, that is, a set of input-output target (desired) examples, related through F, say $\xi^{(m)} \equiv \{ \xi_l , 1 \leq l \leq m\}$, where $\xi \equiv (x, y)$ with $x \in X \subset \Re^3$ and $y \in Y \subset \Re^1$ in our spatial interaction context. The relation F can be generally described by the probability density function defined over the space of input-output pairs $X \otimes Y \subset \Re^4$: $P_F(\xi) = P_F(x)\, P_F(y|x)$, where $P_F(x)$ defines the region of interest in the input space and $P_F(y|x)$ the functional (statistical) relation between the inputs and the outputs. The training set consists of examples drawn independently according to this probability density function.

Learning the training set by a model of the general model $\aleph_{3,\,l,1}$ may be posed as a (non-linear) optimization problem by introducing a quality measure of the approximation of the desired relation F by the mapping F_ω realized by the network (White 1989, Gyer 1992). In this study the additive error function for a set S of examples

$$E(S) \equiv E(\xi(S)\,|\,\omega) = \sum_{l \in S} \mathbf{e}(y_l \mid x_l, \omega) = \frac{1}{2}\sum_{l \in S}(y_l - \hat{y}_l)^2 \qquad (11)$$

has been chosen which measures the dissimilarity between F and F_ω on the restricted domain covered by a set S of input-output target data ($l < m$). The error function $\mathbf{e}(y|x, \omega)$ is a distance measure on $\Re^1$ between the target output (signal) y and the actual output $\hat{y}$ of the network on the given input x. The minimization of E over the network's configuration space is called the training process. The task of learning, however, is to minimize that error for all possible examples related through F, namely, to generalize (Levin, Tishby and Solla 1990).

To accomplish this task, we utilize the *error back propagating technique* of Rumelhart, Hinton and Williams (1986), a recursive learning procedure using a gradient search to minimize (11). Learning is carried out by iteratively adjusting the network parameters. This learning process is repeated until the network responds for each input vector x_l with an output vector $\hat{y}_l$ that is sufficiently close to the target one y_l

Error backpropagating has three major components. In the first component, an input vector $x_l = (x_{1l}, x_{2l}, x_{3l})$ is presented which leads via the forward pass to the activation of the network as a whole. This generates a difference (error) between the output of the network and the desired output. The second component computes the error factor (signal)

for the output unit and propagates this factor successively back through the network (error backward pass). The third component computes the changes for the connection weights and the biases, and updates the parameters. This process continues until the network reaches a satisfactory level of performance.

Error back propagation training may be performed *on-line* (i.e. after each presentation of an input signal; on-line learning) or *off-line* (i.e. after a set S of input presentations, off-line learning). We used the off-line learning mode, more precisely *epoch training* with epoch size of 20 input signals presented in random order (stochastic approximation) and parameter updates following the momentum update rule defined as (Rumelhart, Hinton and Williams 1986):

$$\Delta W_{k, i_k, i_{k+1}}(t+1) = -\eta \frac{\partial E(S)}{\partial W_{k, i_k, i_{k+1}}} + \gamma \Delta W_{k, i_k, i_{k+1}}(t) \qquad (12)$$

where $\partial E(S) / \partial W_{k, i_k, i_{k+1}}$ denotes the partial derivative of E(S) with respect to $W_{k, i_k, i_{k+1}}$, (the error gradient), η is the (constant) learning rate, γ with $0 \leq \gamma < 1$ (the so-called momentum factor) the relative contribution of the previous change of the parameter, and t represents the number of epochs, i.e. the number of times the network has been through a set S of 20 randomly chosen cases, after which the parameters are updated.

It is not easy to choose appropriate values for the learning rate parameter η and the momentum parameter γ for a problem at hand. A learning rate that is too large may cause the model to oscillate, and therby to slow or prevent the network's convergence. The momentum term is much like the learning rate in that it is peculiar to specific error surface contours. The momentum term, if it overwhelms the learning rate, can make the system less sensitive to local changes. No rule for selecting optimal values for η and γ exists, although specific values are sometimes suggested in the literature. Experiments showed that it was best to use a small learning rate and a larger momentum term in our application setting. In all training cases, the learning rate parameter, η, was set to 0.15 and momentum, γ, to 0.8.

The backpropagating technique amounts to performing gradient descent on a hyper surface in weight space Ω where at any point in that space the error of performance (11) is the height of the surface. The procedure, however, is not guaranteed to find a global minimum of E since gradient descent may get stuck in (poor) local minima. Thus, it makes sense to take different initial parameter values, and then select the estimate giving the smallest E. Although a number of more complex adaptation algorithms have been proposed to speed convergence (see Widrow and Lehr 1990, Shah, Palmieri and Datum 1992), it seems unlikely that the complex regions formed by two-layer feedforward

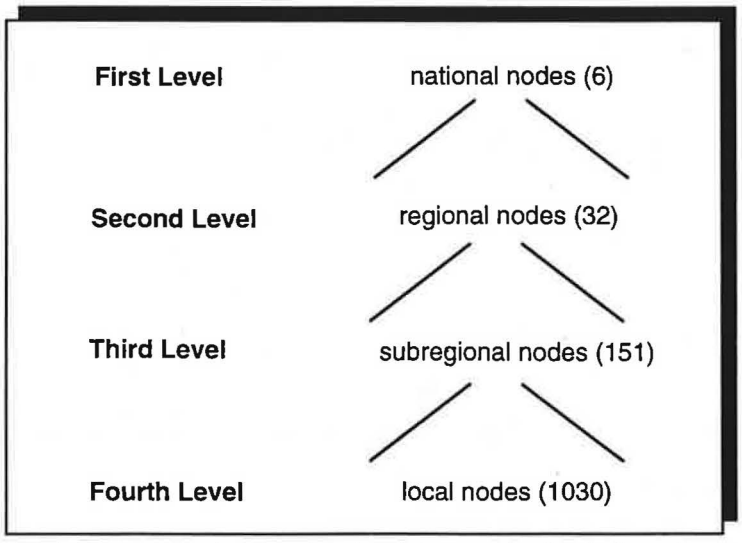
networks can be generated in few epochs when regions are disconnected. As all gradient descent procedures, backpropagation is sensitive to starting points (in the present context, the initial set of coupling strengths and biases in the network). The initial network parameters were drawn at random from an uniform distribution between -0.1 and 0.1. This random initialization prevents the hidden units from acquiring identical weights during training. Five different random parameter initializations (trials) were generated to analyse the robustness of the parameters and the network performance.

3. Experimental Environment and Results

3.1 Data, Training and Testing Results

The telecommunication data used in this study stem from network measurements of carried traffic (facsimile transfers) in Austria in 1991, in terms of erlang, an internationally widely used measure of telecommunication (facsimile transfer) contact intensity, which is defined as the number of phone calls (including facsimile transfers) multiplied by the average length of the call (transfer) divided by the duration of measurement. The data refer to the total telecommunication traffic between the thirty-two telecommunication districts representing the second level of the hierarchical structure of the Austrian telecommunication network (see figures 4 and 5). Due to measurement problems, intraregional traffic (i.e. r = s) is left out of consideration in this study.

Figure 4: The Hierarchical Structure of the Austrian Telecommunication Network



Note: In several cases there are direct lines between regional nodes belonging to different national nodes. The same is true for subregional nodes belonging to different regional nodes.

Figure 5: The Regional System (n=32) for Modelling Interregional Telecommunication in Austria

Thus, the data set for the model building process is made up of a set of n(n-1) = 992 4-tuples (A_r, B_s, D_{rs}; T_{rs}^{obs}) with r,s = 1, ..., 32 and r≠s. The first three components represent the input x = (x_1, ..., x_m, x_l = (x_{l1}, x_{l2}, x_{l3}), l = 1, ..., m), where l labels the individual training pairs, and the last component the desired (target) output (teacher) y = (y_l), displayed in table 1.

Table 1: Input - Target Output Pairs for Training and Testing the Neural Network Model Candidates

Input Components[1]					**Output**[2]
		A_r	B_s	D_{rs}	T_{rs}^{obs}
		(=x_{l1})	(=x_{l2})	(=x_{l3})	(=y_l)
x_1	=	(A_1 ,	B_2 ,	$D_{1,2}$)	y_1 = $T_{1,2}^{obs}$
x_2	=	(A_1 ,	B_3 ,	$D_{1,3}$)	y_2 = $T_{1,3}^{obs}$
x_3	=	(A_1 ,	B_4 ,	$D_{1,4}$)	y_3 = $T_{1,4}^{obs}$
⋮					⋮
x_{31}	=	(A_1 ,	B_{32} ,	$D_{1,32}$)	y_{31} = $T_{1,32}^{obs}$
x_{32}	=	(A_2 ,	B_1 ,	$D_{2,1}$)	y_{32} = $T_{2,1}^{obs}$
x_{33}	=	(A_2 ,	B_3 ,	$D_{2,3}$)	y_{33} = $T_{2,3}^{obs}$
⋮					⋮
x_{961}	=	(A_{32} ,	B_1 ,	$D_{32,1}$)	y_{961} = $T_{32,1}^{obs}$
⋮					⋮
x_{992}	=	(A_{32} ,	B_{31} ,	$D_{32,31}$)	y_{992} = $T_{32,31}^{obs}$

1 A_r represents the potential pool of calls in region r, measured in terms of gross regional product in r (r = 1, ..., 32); B_s represents the potential draw of calls in region s, measured in terms of the gross regional product in s (s = 1, ..., 32);

2 T_{rs}^{obs} denotes the observed telecommunication flow from region r to region s (r≠s), measured in terms of erlang

The input and output signals were preprocessed to logarithmically transformed data scaled into (0,1), partly in order to make the comparison with the gravity model (4) as close as possible, and partly, because experiments showed that it is easier for the model candidates to learn the logarithmically transformed data than the original raw data. The continuous-valued input signals were presented one point at a time to activate the network. The response generated at the output in response to each input was then used to calculate the error with respect to the teacher. The weights and biases were updated after each 20 patterns presented in random order (accumulated parameter correction). In one epoch the network sees each point from the training set exactly once. If the samples are chosen in random order, it also makes the path through weight-space stochastically, allowing wider exploration of the error surface.

3.2 Model Identification

After fixing I_1 and I_3 to 3 and 1 respectively, the identification process involves the determination of I_2 only, the issue of how many hidden units are needed. The choice of the number of hidden units in a feedforward structure design is not an easy task. It is generally problem dependent and often involves considerable engineering judgement. Often trade-offs between training time and modelling performance lead to iterative judgement of the network using simulation. For a given problem, the design of an adequately sized hidden layer is often not very obvious. Intuition suggests that 'the more the better' rule might be used to guide sizing the hidden layer, since the number of hidden units controls the model's flexibility. But very large sized hidden layers may become counterproductive. Too many free network parameters will allow the network to fit the training data arbitrarily closely, but will not necessarily lead to optimal predictions. There is no general procedure to determine the optimal size of the hidden layer for a particular task. A rule-of-thumb often cited is that the number of weights should be less than one tenth of the number of training patterns (Weigend, Rumelhart, and Huberman, 1991). With a relatively small training set, this constraint is too restrictive.

The training set used in the identification stage of the model building process consists one-half of the available data, i.e. 496 patterns. However, this is not a hard rule. To determine "I" experimentally, i.e. the number of hidden units required, we considered 10 model candidates: (3:5:1), (3:10:1), (3:15:1), (3:20:1), (3:25:1), (3:30:1), (3:35:1), (3:40:1), (3:45:1), and (3:50:1). Each model candidate was trained on the training set for 3000 epochs. The initial parameters were drawn at random from an uniform distribution between -0.1 and 0.1. Five different random initializations were used to analyse variations due to different random initial conditions. The weights were updated after each 20 input signals as defined by equation (12), presented in random order (stochastic

approximation). The model identification process is based on the networks' overall performance, specified in terms of two measures. Performance of each model candidate was taken to be the average over a set of five performance values obtained from the five trials.

The first performance measure used is the average relative variance ARV(S) of a set S of patterns, which is widely used in the neural network literature (see Weigend, Rumelhart and Huberman 1991) defined as

$$\text{ARV}(S) = \frac{\sum_{l \in S} (y_l - \hat{y}_l)^2}{\sum_{l \in S} (y_l - \bar{y})^2} = \frac{1}{\hat{\sigma}^2} \frac{1}{N_S} \sum_{l \in S} (y_l - \hat{y}_l)^2 \qquad (13)$$

where y_l denotes the target value and $\hat{y}_l$ the actual network value, $\bar{y}$ the average over the 20 desired values in S. The averaging, i.e. division by N_S (the number of patterns in set S (= epoch), $N_S = 20$) makes ARV(S) independent of the size of the set. The division by the estimated variance $\hat{\sigma}^2$ of the data removes the dependence on the dynamic range of the data. This implies that if the estimated mean of the observed data would be taken as predictor, ARV(S) would be equal to 1 (Weigend, Rumelhart and Huberman 1991). In this study the variances of the individual sets S associated with the different epochs differ only slightly. Thus it appears to be reasonable to always use the variance of the entire data record $\hat{\sigma}^2 = \sigma^2_{all} = 3.112$ as a proxy.

A value of ARV(S) = 0.1 corresponds, thus, to an average absolute quadratic error of $\text{ARV}(S) \cdot \sigma^2_{all} = 0.1 \cdot 3.112 = 0.3112 \approx 0.9682^2$. The alternative would have been to normalize each individual set S by its own variance.

The second performance measure used in this study is the coefficient of determination $R^2(S)$, a widely used goodness-of-fit measure in spatial interaction modelling (Fotheringham and O'Kelly 1989). This measure is defined as

$$R^2(S) = \frac{\sum_{l \in S} (\hat{y}_l - \hat{\bar{y}})^2}{\sum_{l \in S} (y_l - \bar{y})^2} = \frac{1}{\hat{\sigma}^2} \frac{1}{N_S} \sum_{l \in S} (\hat{y}_l - \hat{\bar{y}})^2 \qquad (14)$$

with $0 \leq R^2 \leq 1$ and $\hat{\bar{y}}$ denoting the average over the 20 predicted values in S.

Table 2: Model Identification: Performance of Selected Model Candidates from $\aleph_{3,I,1}$

Neural Net Model	Number of Parameters	Trial[1]	Performance Measures[2] ARV	R^2
3:5:1	26	1	0.0855	0.6163
		2	0.0898	0.6313
		3	0.0825	0.7585
		4	0.0803	0.7636
		5	0.0793	0.7658
		Av. Performance	0.0835(0.0043)[3]	0.7071(0.0763)[3]
3:10:1	51	1	0.0813	0.7613
		2	0.0793	0.7710
		3	0.0810	0.7698
		4	0.0788	0.7725
		5	0.0804	0.7685
		Av. Performance	0.0802(0.0011)[3]	0.7686(0.0041)[3]
3:15:1	76	1	0.0800	0.7685
		2	0.0799	0.7685
		3	0.0791	0.7651
		4	0.0800	0.7650
		5	0.0814	0.7660
		Av. Performance	0.0801(0.0008)[3]	0.7666(0.0018)[3]
3:20:1	101	1	0.0773	0.7751
		2	0.0737	0.7833
		3	0.0778	0.7757
		4	0.0769	0.7775
		5	0.0778	0.7749
		Av. Performance	0.0767(0.0017)[3]	0.7773(0.0035)[3]
3:25:1	126	1	0.0788	0.7727
		2	0.0794	0.7722
		3	0.0775	0.7767
		4	0.0775	0.7757
		5	0.07751	0.7763
		Av. Performance	0.0781(0.0009)[3]	0.7747(0.0021)[3]
3:30:1	151	1	0.0774	0.7755
		2	0.0771	0.7786
		3	0.0781	0.7770
		4	0.0766	0.7820
		5	0.0787	0.7766
		Av. Performance	0.0776(0.0008)[3]	0.7780(0.0025)[3]
3:35:1	176	1	0.0847	0.7613
		2	0.0808	0.7712
		3	0.0811	0.7652
		4	0.0820	0.7666
		5	0.0794	0.7690
		Av. Performance	0.0816(0.0020)[3]	0.7667(0.0038)[3]
3:40:1	201	1	0.0766	0.7839
		2	0.0783	0.7763
		3	0.0764	0.7779
		4	0.0795	0.7740
		5	0.0798	0.7739
		Av. Performance	0.0781(0.0016)[3]	0.7772(0.0041)[3]
3:45:1	226	1	0.0871	0.6222
		2	0.0833	0.6377
		3	0.0867	0.6227
		4	0.0852	0.6299
		5	0.0869	0.6224
		Av. Performance	0.0858(0.0016)[3]	0.6270(0.0068)[3]
3:50:1	251	1	0.0868	0.6420
		2	0.0866	0.6532
		3	0.0869	0.6365
		4	0.0868	0.6421
		5	0.0870	0.6310
		Av. Performance	0.0868(0.0002)[3]	0.6400(0.0082)[3]
Conventional Model	3			
		1	0.0880	0.6803
		2	0.0813	0.6080
		3	0.0855	0.6729
		4	0.0848	0.6930
		5	0.0870	0.6732
		Av. Performance	0.0863(0.0032)[3]	0.6768(0.0080)[3]

1 The trials differ in initial conditions as well as in the random sequence of the input signals; epoch size of 20 patterns presented in random order, the training set consists of 496 patterns (50 percent of the available data).

2 ARV: (average relative variance) as defined by (13); R^2 (coefficient of determination) as defined by (14); average performance is the mean over the given performance values; the training set consists of 496 points, conventional model: Gravity model (4)

3 Standard deviation

The resulting fits of the model candidates, in terms of ARV and R^2, are reported in table 2 for each of the five trials. Compared to the benchmark model's results, we find strikingly good fits for seven model candidates: (3:10:1), (3:15:1), (3:20:1), (3;25:1), (3:30:1), (3:35:1) and (3:40:1). The best average R^2-performance is achieved by the (3:30:1) model with $R^2 = 0.7780$ followed by (3:20:1) with 0.7773 and (3:40:1) with 0.7772. In terms of the average ARV-performance a slightly different ranking is obtained. (3:20:1) performs slightly better than the (3:30:1) model, followed by the (3:40:1) and (3:25:1) models. The variation in ARV- and R^2-performance due to initial conditions is very moderate.

It is well known, that there can be little connection between the training error, restricted to the training set, and the network's ability to generalize outside of that set. The (3:30:1) model unequivocally shows a better generalization capability than the (3:20:1) model. The (3:30:1) network outperforms the (3:20:1) network on the testing set, in terms of the average ARV-performance (0.4131 compared to 0.4198) and the average R^2-performance (0.5935 compared to 0.5827). In addition, (3:30:1) tends to be less sensitive to initial conditions. Thus, (3:30:1) rather than (3:20:1) has been chosen as the appropriate model for the problem at hand.

3.3 Model Estimation and the Overfitting Problem

Whereas the stage of identification was concerned with identifying an appropriate model from the family $\aleph_{3,\,l,1}$, the stage of parameter estimation is devoted to the determination of the magnitude and sign of the parameters of the (3:30:1) network model. A serious problem in model estimation is the problem of overfitting which is particularly serious for noisy real world data of limited record length. As opposed to computer generated data, the noise level in any real world context is not known a priori.

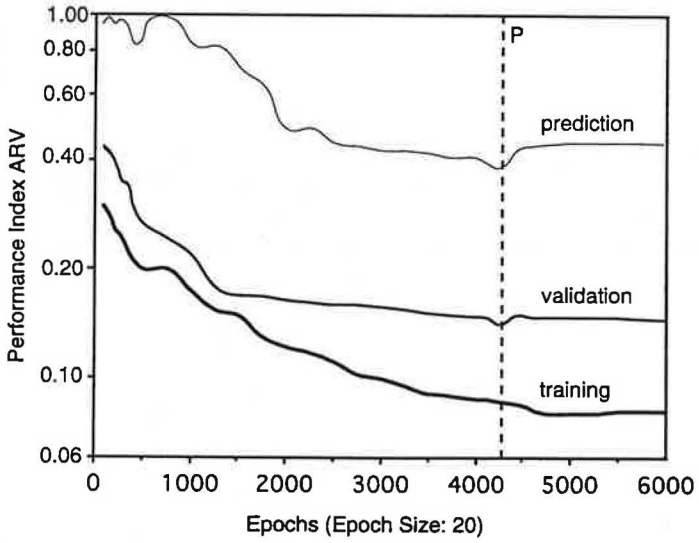
If the network mapping F_ω fits the training data exactly, the capability of the network to generalize, that is to generate an acceptable output when a novel input signal is applied to the network, will often be poor. This arises from the rapid oscillations in the network function that are generally needed to fit the noisy training data. To improve the generalization capability of the network model, it is necessary for the network mapping to represent the underlying trends in the data, rather than fitting all of the fine details of the data set (Bishop 1991).

Several strategies for handling the overfitting problem have been devised (Hecht-Nielsen 1990, Bishop 1991, Weigend, Rumelhart and Huberman 1991). One possibility is to take a subset of input vectors from the training data. The subset may be chosen randomly or by a more systematic elimination procedure. We used the random method of cross-validating

to detect when overfitting occurs. It is crucial to split the whole available data set of 992 patterns into three separate subsets: a training (estimation) set, a validation test set and a testing (prediction) set. While the training set is directly used for training the neural network model, the validation test set is used only for the evaluation process, i.e. for determining when to stop the training process. The testing or prediction set is strictly set apart and never used in the estimation stage. The training set-validation set pair consists of two thirds of the available data. One quarter of these data (i.e. 148 patterns) have been used for validation process. It is crucial that the validation test set is only utilized to detect a statistically proper stopping point of the training process of the (3:30:1) model.

Figure 6 shows the performance of the (3:30:1) neural network model as a function of training time in epochs with an epoch size of 20 patterns. The average relative variances are given for the training set, the validation set and the prediction set. The ARV-error of the model measured using the training set continuously decreases and seems to level out after 5,000 epochs. This is what one expects for a model of $\aleph_{I_1, I_2, I_3}$. The validation test set error as shown in figure 6 decreases first, after 1,500 epochs only at a moderate rate until 4,250 epochs, then slightly increases and tends to approach an asymptotic value. If we assume that the error curve of the (3.30:1) model tested against the entire infinite set of possible patterns would be approximately the same as that of the validation set curve (Hecht-Nielsen 1990), which is only a crudely correct assumption, then clearly we would like to stop training when this curves arrives at its minimum. The minimum is reached after 4,250 epochs. At this stopping point, P, the model is used for prediction. In the specific choice of a training set-validation set combination shown in figure 6, the fitting of the noise of the training set occurs to have only a little effect on the error of the validation set, which is also reflected in the prediction set curve.

Figure 6: Training, Validation and Prediction Set Curves of the (3:30:1) Network Model as a Function of Training Time in Epochs (the vertical line P indicates the stopping point)



To get a feeling for the effect of the sampling error induced by using a specific training set-validation set combination, we analysed several such pairs. Because the telecommunication data set at hand is rather small, different pairs of training and validation test sets each, drawn randomly from the available data set, did lead to different ARV-results differing by factors up to two. These variations are rather large compared to the small variations due to different random initial parameters. Consequently, the statistical procedure of cross-validating is somewhat unsatisfactory.

3.4 Neural Net and Gravity Model Predictions

The ultimate goal of the (3:30:1) neural net model is to predict interregional telecommunication traffic flows or in other words to determine how well the network learned to approximate the unknown input-output function for arbitrary values of x. Thus, we briefly discuss the predictive quality of the neural net model and compare it with that of the gravity model. It is important to note that the network model is no longer allowed to adapt while it is being tested. To assess the prediction performance, we primarily use the average relative variance ARV as defined in equation (13) and the coefficient of determination R^2 as defined in equation (14).

Table 3: Testing Performance of the (3:30:1) Neural Net and the Gravity Model*

	(3:30:1) Neural Net Model		Conventional Model	
	ARV	R^2	ARV	R^2
Prediction (Testing)				
Trial 1	0.4063	0.5937	0.4630	0.5431
Trial 2	0.4057	0.5942	0.4611	0.5390
Trial 3	0.4063	0.5938	0.4582	0.5422
Trial 4	0.4077	0.5923	0.4761	0.5310
Trial 5	0.4064	0.5934	0.4892	0.5290
Average Performance	0.4131	0.5935	0.4695	0.5353
(Standard Deviation)	(0.0155)	(0.0007)	(0.0130)	(0.0068)

* The trials differ in initial conditions as well as in the random sequence of the input signals;
ARV: (average relative variance) as defined by (13); R^2 (coefficient of determination) as defined by (14); average performance is the mean over the given performance values; the testing set consists of 348 points, conventional model: gravity model (4)

Table 4: Prediction Accuracy of the (3:30:1) Neural Net and the Gravity Model: Some Selected Results

T_{rs}	Observation	Model Predictions	
		Neural Net	Conventional
$T_{2.17}$	69.7279	44.2051	57.8402
$T_{3.30}$	0.2113	0.1219	0.7010
$T_{4.7}$	12.1801	11.5071	13.8977
$T_{5.21}$	1.7500	1.6858	2.1387
$T_{11.23}$	0.4314	0.4695	0.5587
$T_{13.7}$	1.4137	1.8286	2.3708
$T_{16.9}$	10.4940	10.8594	19.4737
$T_{18.4}$	21.9815	20.5638	23.3007
$T_{20.11}$	4.4647	3.7864	3.4129
$T_{28.16}$	1.5310	1.6492	1.6535
$T_{29.18}$	15.082	19.7773	30.9833

Table 3 reports the prediction performance of the (3:30:1) network on the testing set in terms of ARV and R^2 averaged over the five trials. In order to make the comparison with the gravity model as close as possible, this model was estimated (tested) with the same data seen by the neural net model during training (testing). As can be seen by comparing the ARV- and R^2-values, the neural network leads to a somewhat higher prediction performance in all trials. The average prediction quality measured in terms of ARV and R^2 is 0.4131 and 0.5932 respectively, compared to 0.4695 (ARV) and 0.5353 (R^2) for the gravity model. The prediction quality is rather stable over the different trials. In table 4 the prediction accuracy achieved by the two alternative interregional teletraffic models is exemplified by a systematic sample of T_{rs}-values.

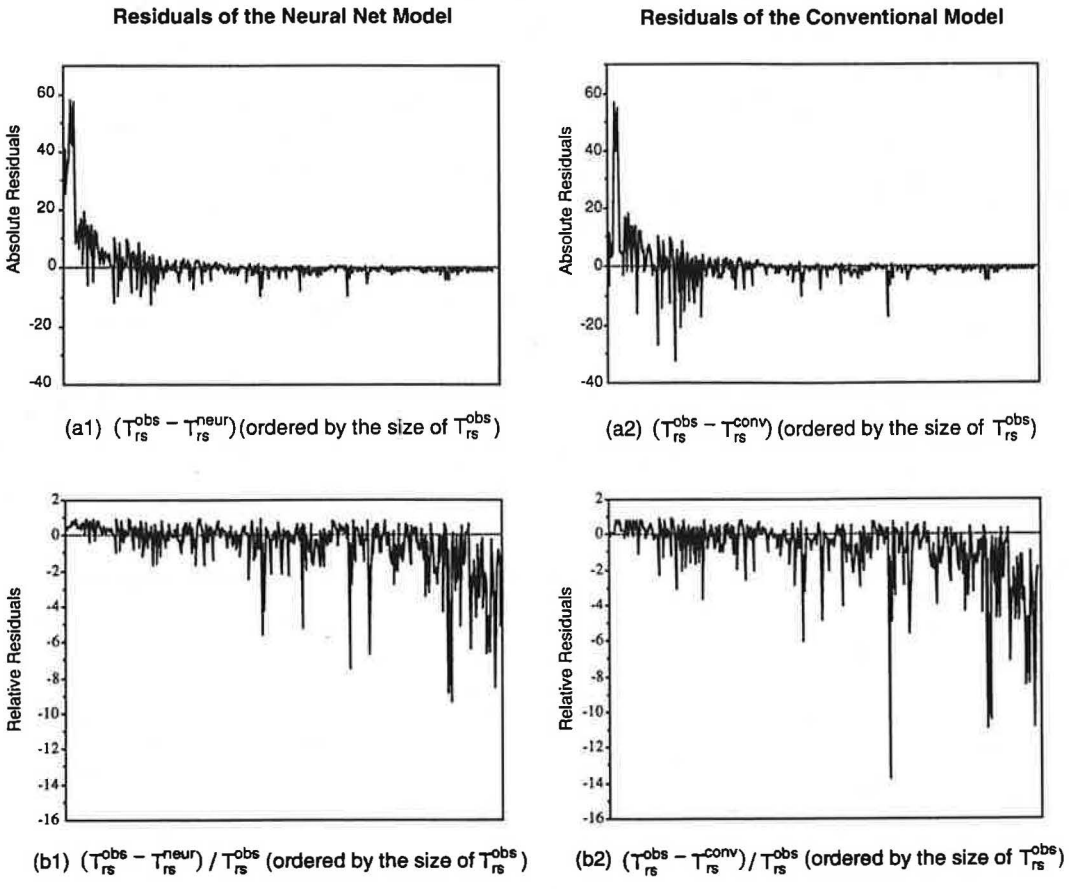
One means of further investigating the predictive power is the use of residual analysis. Figure 7 graphically displays in terms of

(a) the absolute residuals of the individual flows $(T_{rs}^{obs} - T_{rs}^{conv})$compared to $(T_{rs}^{obs} - T_{rs}^{neur})$,

(b) the relative residuals of the individual flows $(T_{rs}^{obs} - T_{rs}^{neur}) / T_{rs}^{obs}$ and $(T_{rs}^{obs} - T_{rs}^{conv}) / T_{rs}^{obs}$,

where both absolute and relative residuals are ordered by the size of the T_{rs}^{obs} flows.

Figure 7: Residuals of the 3:30:1 Neural Net and the Gravity Model Predictions



The main conclusions from this analysis can be summarized as follows:

- *First*, both models show a tendency to underpredict larger flows, especially those between dominant (provincial capital) regions. Examples include flows from Vienna to Klagenfurt, Salzburg to Linz, and Salzburg to Klagenfurt. They also underpredict flows involving dominant regions and neighbouring minor regions, such as, for example, the flows from Graz to Hartberg, Innsbruck to Reutte, Ried/Innkreis to Salzburg, and Wolfsberg to Klagenfurt. The neural network model underpredicts 63 out of 87 flows in the largest quartile, compared to 51 gravity model underpredictions.

- *Second*, and in addition, the two alternate models share a tendency to overpredict smaller flows representing generally flows between minor regions further apart from each other. This is evidenced by 67 neural network and 75 gravity model overpredictions in the smallest quartile.

- *Third*, the neural network model and the conventional model show a relatively similar pattern of residuals. Despite this similarity, however, the neural network produces more accurate predictions for 188 of the 348 flows considered in the testing stage. This is also indicated by the average (that is, root-mean-squared) error of 20.52 percent, while the corresponding figure amounts to 24.56 percent for the gravity model. Many of the differences are small. But in some cases, the flows are replicated substantially more accurately by the neural network model. Figure 7 (b2) provides evidence that flow errors of the gravity model may occasionally exceed even over 100 percent (4 cases).

In summary, the analysis unequivocally shows that the neural net model outperforms the gravity model in terms of both the ARV and R^2 prediction performance as well as prediction accuracy, but to a lesser degree than previously expected. One reason for this might be that the validation set was relatively small and the statistical method of cross-validating the network against validation data did not indicate unequivocally the stopping point, another reason being the training procedure. In addition, there is an indication that memorization of the training set is interfering with the ability of the (3:30:1) network to generalise. This is an issue for further research.

4. Summary and Conclusions

The primary advantage of the general neural network model set out in this paper over the classical regression approach to spatial interaction modelling lies in the fact that it has a more general functional form than the gravity model can effectively deal with. The neural network model implements a functional input--output relationship that is expressed in terms of a general, modifiable form. This functional form is modified via the adaptive setting of weights by means of the application of the feedback propagating technique and

the additive (mean squared) error function to fit the specific mapping which is approximated. It may be viewed as a non-linear regression function of a quite specific form (White 1989). The methods of non-linear regression analysis evidently resemble those of neural network modelling. But none, if any, individual statistical regression function procedures have been developed as the neural network model presented in this paper.

For sigmoid processing units, relatively good local solutions were obtained for the interregional (3:30:1) neural net telecommunication model in all trials with different initial random network parameters. Sensitivity to initial conditions was rather moderate. This might have been mainly due to choosing relatively small initial parameter values, a small learning rate parameter η, a relatively large momentum rate parameter γ, a relatively large hidden layer and the logistic rather than the hyperbolic tangent transfer function for the problem at hand. On the noisy real-world telecommunication data of limited record length the analysis illustrated the superiority of the neural network model over the classical regression approach. But neural network modelling requires a procedure to deal with the problem of overfitting. The statistical method of cross-validating the network against a reserved set of validation data is certainly unsatisfactory at least due to two reasons. First, the results depend on the specific training set-validation set pair used for deciding when to stop training. Second, it is not always completely clear from the ARV of the validation set when the training process should be stopped to avoid overfitting. The issue of overfitting deserves further research efforts in future. One alternative strategy to tackle this problem might be the strategy of weight elimination suggested by Weigend, Rumelhart and Huberman (1991). This strategy involves the extension of the gradient method by adding a complexity term to the error (cost) function which associates a cost element with each connection.

Neural networks have an important role to play in geography and regional science not only in the application domain of spatial interaction modelling, but also in exploratory spatial data analysis. They can principally be used to cope with problems such as very large volumes of spatial data, missing data, noisy data and fuzzy information for which conventional statistical techniques may be inappropriate or cumbersome to use. It is hoped that the methodology outlined in this paper will make the study of neural networks more accessible to the regional science community and will stimulate further research in this new and promising field.

Acknowledgements: *The quality of this paper substantially improved by the comments provided by three anonymous reviewers on a first submission of the paper; we gratefully appreciate their comments. The second author wishes to acknowledge the assistance and computational environment provided when she was a visiting professor at the Vienna University of Economics and Business Administration, Department of Economic and Social Geography. The authors are grateful for the funding provided by the Austrian Fonds zur Foerderung der wissenschaftlichen Forschung (P 09972-TEC) and the US-National Science Foundation (SBR-9300633). Moreover, both authors would like to thank Petra Staufer for her valuable assistance in the preparation of the manuscript and illustrations.*

References

Amari, Shun-Ichi (1990): Mathematical foundations of neurocomputing, **Proceedings of the IEEE** 78, pp. 1443-1463.

Barron, Andrew R. (1993): Universal approximation bounds for superpositions of a sigmoidal function, **Ieee Transactions on Information Theory** 39, pp. 930 -945.

Bishop, Chris (1991): Improving the generalization properties of radial basis function neural networks, **Neural Computation** 3, pp. 579-588.

Cybenko, G. (1989): Approximation by superpositions of sigmoidal function, **Mathematics of Control, Signals, and Systems** 2, pp. 303-314.

Fischer, Manfred M. (1993): Travel demand, in Polak, Jacob and Heertje, Arnold (eds.): **European Transportation Economics**, pp. 6-32. Oxford (UK) and Cambridge (MA): Basil Blackwell.

Fischer, Manfred M. and Gopal, Sucharita (1993): Neurocomputing - a new paradigm for geographic information processing, **Environment and Planning A** 25, pp. 757-760.

Fischer, Manfred M., Essetzbichler, Jürgen, Gassler, Helmut and Trichtl, Gerhard (1992): Interregional and international telephone communication. Aggregate traffic models and empirical evidence for Austria, **Sistemi Urbani** (in press).

Fotheringham, Stewart A., and O'Kelly, Morton E. (1989): **Spatial Interaction Models: Formulations and Applications.** Dordrecht, Boston and Lodon: Kluwer.

Funahashi, Ken-Ichi (1989): On the approximate realization of continous mappings by neural networks, **Neural Networks** 2, pp. 183-192.

Gillespie, Andrew and Williams, Howard (1988): Telecommunications and the reconstruction of regional comparative advantage, **Environment and Planning A** 20, pp. 1311-1321.

Guldmann, Jean-Michel (1992): Modeling residential and business telecommunication flows: A regional point-to-point approach, **Geographical Analysis** 24, pp. 121-141.

Gyer, Maurice S. (1992): Adjuncts and alternatives to neural networks for supervised classification, **IEEE Transactions on Systems, Man, and Cybernetics** 22 , pp. 35-47.

Hecht-Nielsen, Robert (1990): **Neurocomputing.** Reading (MA): Addison-Wesley.

Hornik, Kurt, Stinchcombe, Maxwell and White, Halbert (1989): Multilayer feedforward networks are universal approximators, **Neural Network** 2, pp. 359-366.

Levin, Esther, Tishby, Naftali, and Solla, Sara A. (1990): A statistical approach to learning and generalization in layered neural networks, **Proceedings of the IEEE** 78, pp. 1568-1575.

Openshaw, Stan (1992): Modelling spatial interaction using a neural net, in Fischer, Manfred M. and Nijkamp, Peter (eds): **Geographic Information Systems, Spatial Modelling and Policy Evaluation,** pp. 147-164. Berlin: Springer.

Pacey, P. L. (1983): Long distance demand: A point-to-point model, **Southern Economic Journal** 49, pp. 1094-1107.

Rietveld, Piet and Janssen, Leon (1990): Telephone calls and communication barriers. The case of the Netherlands, **The Annals of Regional Science** 24, pp. 307-318.

Rossera, Fabio (1990): Discontinuities and barriers in communications. The case of Swiss communities of different language, **The Annals of Regional Science** 24, pp. 319-336.

Rummelhart, David E., Hinton, Geoffrey E. and Williams, Ronald J. (1986): Learning internal representations by error propagation, in Rummelhart, David E., McClelland, James L. and the PDP Research Group (eds.) (1986): **Parallel Distributed Processing. Explorations in the Microstructures of Cognition. Volume 1: Foundations**, pp. 318-362. Cambridge (Ma.) and London (England): The MIT Press.

Salomon, Ilan (1986): Telecommunications and travel relationships: A review, **Transportation Research** 3, pp. 223-238.

Shah, Samir, Palmieri, Francesco and Datum, Michael (1992): Optimal filtering algorithms for fast learning in feedforward neural networks, **Neural Networks** 5, pp. 779-787.

Weigend, Andreas S., Rumelhart, David E. and Huberman, Bernardo A. (1991): Back-propagation, weight-elimination and time series prediction, in Touretzki, David S., Elman, Jeffrey L., Sejnowski, Terrence J., and Hinton, Geoffrey E. (eds.): **Connectionist Models. Proceedings of the 1990 Summer School**, pp.105-116. San Mateo (CA), Morgan Kaufmann Publishers.

White, Halbert (1989): Some asymptotic results for learning in single hidden-layer feedforward network models, **Journal of American Statistical Association** 84, pp. 1003-1013.

Widrow, Bernard , and Lehr, Michael A. (1990): 30 years of adaptive neural networks: Perceptron, Madaline, and Backpropagation, **Proceedings of the IEEE** 78, pp. 1415-1447.

Appendix: Parameters of the (3:30:1)-Interregional Teletraffic Neural Network Model

The (3:30:1) network was trained for 4,250 epochs with a learning rate $\eta = 0.15$ and a momentum $\gamma = 0.8$. The weights were updated after each 20 patterns presented in random order. The connection weights and biases of the network are given below in Table A1. When simulated serially on a SparcStation 1+, the training of the 3:30:1 takes less than 6 CPU-minutes for 4,250 epochs. Once the network parameters have been determined, predictions are extremely fast. The parameter estimates of the gravity model (4) are explicitly given in Table A2.

Table A1: Parameter Estimates W_{1,i_1,i_2}; W_{2,i_2,i_3} of the (3:30:1) Neural Net Interregional Teletraffic Model (Trial 3 see Table 3)

Weights W_{1,i_1,i_2} from	Input Unit $i_1 = 1$		Input Unit $i_1 = 2$		Input Unit $i_1 = 3$		Bias Unit	
	Start	Final	Start	Final	Start	Final	Start	Final
to Hidden Unit 1	0.0073	0.1045	0.0160	0.1022	0.0662	-0.0839	0.0680	-0.0256
to Hidden Unit 2	-0.0985	-0.5126	-0.0213	-0.3944	-0.0116	0.2485	0.0164	-0.0347
to Hidden Unit 3	0.0022	-0.2100	-0.0488	-0.2391	-0.0134	0.0745	0.0829	0.0070
to Hidden Unit 4	-0.0150	-0.1397	0.0981	-0.0181	-0.0156	0.0045	-0.0787	-0.1552
to Hidden Unit 5	0.0005	-0.1220	-0.0002	-0.1118	-0.0246	-0.0011	-0.0719	-0.1457
to Hidden Unit 6	-0.0818	-0.1307	0.0509	0.0048	-0.0890	-0.1207	-0.0439	-0.1206
to Hidden Unit 7	0.0504	0.4725	-0.0079	0.3707	-0.0081	-0.4110	0.0757	-0.0372
to Hidden Unit 8	-0.0064	-0.3705	-0.0871	-0.4116	0.0868	0.2903	0.0823	0.0098
to Hidden Unit 9	0.0225	-0.0459	0.0878	0.0233	0.0781	0.0511	-0.0063	-0.0916
to Hidden Unit 10	0.0251	0.2904	0.0423	0.2780	0.0656	-0.2065	-0.074	-0.1678
to Hidden Unit 11	-0.0201	0.3393	0.0859	0.4052	0.0406	-0.3144	0.0581	-0.0525
to Hidden Unit 12	0.0269	0.3442	0.0993	0.3788	0.0072	-0.3166	0.0331	-0.0728
to Hidden Unit 13	-0.0265	0.4407	0.0603	0.4770	-0.0953	-0.5211	0.0510	-0.0506
to Hidden Unit 14	0.0729	0.0205	0.0454	-0.0049	0.0812	0.0398	0.0411	-0.0472
to Hidden Unit 15	0.0935	0.7405	0.0783	0.6618	0.0382	-0.5259	-0.0865	-0.1940
to Hidden Unit 16	0.0536	0.2845	0.0461	0.2475	-0.0826	-0.3396	0.0754	-0.0235
to Hidden Unit 17	0.0714	0.4191	0.0007	0.3104	-0.0193	-0.3629	0.0441	-0.0606
to Hidden Unit 18	-0.0288	0.1483	-0.0149	0.1440	0.0210	-0.1846	0.0520	-0.0414
to Hidden Unit 19	-0.0737	-0.1998	0.0151	-0.0986	-0.0486	-0.0233	0.0495	-0.0276
to Hidden Unit 20	0.0780	-0.0512	-0.0344	-0.1507	0.0945	0.1178	-0.0229	-0.1031
to Hidden Unit 21	-0.0014	0.3241	0.0135	0.3032	-0.0559	-0.3749	0.0369	-0.0601
to Hidden Unit 22	-0.0408	-0.0891	-0.0593	-0.1013	-0.0236	-0.0540	0.0182	-0.0607
to Hidden Unit 23	-0.0835	-0.4889	-0.0225	-0.3865	0.0454	0.2853	0.0784	0.0126
to Hidden Unit 24	-0.0288	0.2170	-0.0195	0.1998	-0.0629	-0.3207	0.0732	-0.0214
to Hidden Unit 25	0.0596	0.1735	-0.0745	0.0264	-0.0183	-0.1775	0.0304	-0.0589
to Hidden Unit 26	-0.0787	-0.4814	-0.0585	-0.4193	0.0566	0.3126	-0.0502	-0.0976
to Hidden Unit 27	-0.0383	-0.1429	-0.0089	-0.1026	-0.0031	0.0053	0.0366	-0.0425
to Hidden Unit 28	-0.0138	-0.2159	0.0405	-0.1424	0.0372	0.1190	-0.0453	-0.1182
to Hidden Unit 29	0.0619	-0.1300	-0.0393	-0.2139	-0.0700	0.0060	-0.0849	-0.1535
to Hidden Unit 30	-0.0251	0.1863	0.0369	0.2245	0.0070	-0.2300	0.0805	-0.0181

Weights W_{2,i_2,i_3} to	Output Unit $i_3 = 1$	
	Start	Final
from Hidden Unit 1	-0.0590	-0.2391
from Hidden Unit 2	0.0410	0.1182
from Hidden Unit 3	-0.0448	-0.7077
from Hidden Unit 4	-0.0297	-0.3492
from Hidden Unit 5	-0.0787	-0.1551
from Hidden Unit 6	-0.0362	-0.1870
from Hidden Unit 7	-0.0206	-0.0704
from Hidden Unit 8	0.0929	0.6580
from Hidden Unit 9	-0.0103	-0.6427
from Hidden Unit 10	-0.0466	-0.0925
from Hidden Unit 11	0.0603	0.3939
from Hidden Unit 12	0.0802	0.5462
from Hidden Unit 13	0.0234	0.5274
from Hidden Unit 14	0.0795	0.7475
from Hidden Unit 15	-0.0460	-0.0629
from Hidden Unit 16	0.0997	10.0524
from Hidden Unit 17	-0.0166	0.4209
from Hidden Unit 18	0.0517	0.5621
from Hidden Unit 19	0.0834	0.2221
from Hidden Unit 20	-0.0118	-0.2163
from Hidden Unit 21	-0.0212	-0.2186
from Hidden Unit 22	0.0658	0.5064
from Hidden Unit 23	0.0395	-0.1266
from Hidden Unit 24	-0.0286	-0.7038
from Hidden Unit 25	0.0755	0.3575
from Hidden Unit 26	0.0365	0.1629
from Hidden Unit 27	-0.0098	-0.7256
from Hidden Unit 28	0.0029	-0.1940
from Hidden Unit 29	-0.0502	-0.3205
from Hidden Unit 30	-0.0834	-0.2587
from Bias Unit $i_2 = 31$	-0..590	-0.2357
R^2	0.1022	0.7161
R^2 adjusted	0.1021	0.7159

Table A2: Parameter Estimates of the Gravity Model

	Gravity Model (4)	
Constant	-19.5976	
$\hat{\alpha}_1$	0.7297	(52.410)*
$\hat{\alpha}_2$	0.7035	(49.366)*
$\hat{\alpha}_3$	-0.7156	(-31.650)*
R^2 (adjusted)	0.5390	(0.5387)

* t-values